12. Listas, pilas y colas

Programación - 1º DAM

Luis del Moral Martínez

versión 20.10

Bajo licencia CC BY-NC-SA 4.0

Contenidos del tema

1. Listas enlazadas
2. Pilas
3. Colas

1. Listas enlazadas

Concepto de lista enlazada

- Una **lista enlazada** es una colección de un número indeterminado de elementos
- Cada elemento tiene diversos componentes:
  - Campos, o **atributos**
  - Un **puntero** al siguiente elemento de la lista
- Los elementos se enlazan entre sí usando los punteros
- Se construye usando la memoria dinámica

1. Listas enlazadas

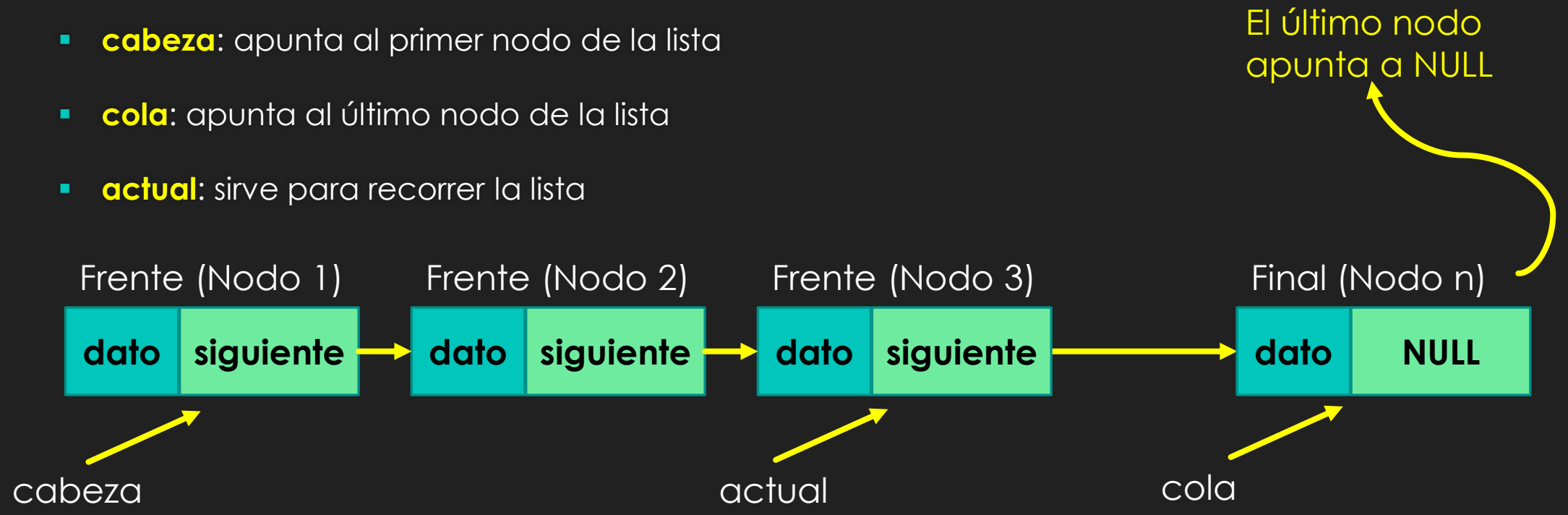
Tipos de listas enlazadas

- Existen los siguientes tipos de listas enlazadas:
  - **Listas simplemente enlazadas**: cada nodo contiene un enlace con el siguiente nodo
  - **Listas doblemente enlazadas**: se mantiene también un enlace con el nodo anterior
  - **Lista circular simplemente enlazada**: el último elemento de la lista se enlaza con el primero
  - **Lista circular doblemente enlazada**: igual que la anterior, pero doblemente enlazada

1. Listas enlazadas

Representación de una lista enlazada

- En la estructura de la lista tenemos, además, tres punteros adicionales:
  - **cabeza**: apunta al primer nodo de la lista
  - **cola**: apunta al último nodo de la lista
  - **actual**: sirve para recorrer la lista



1. Listas enlazadas

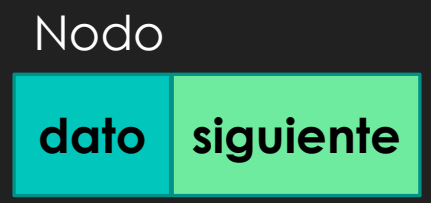
Operaciones en listas enlazadas

- En una lista enlazada se pueden efectuar las siguientes operaciones:
  - Inicialización o creación, declaración de los nodos
  - Insertar elementos en una lista
  - Eliminar elementos de una lista (comprobando que existen previamente)
  - Recorrer una lista
  - Comprobar si la lista está vacía

1. Listas enlazadas

Declaración y creación de un nodo

- Los nodos de la lista se crean usando una estructura de datos

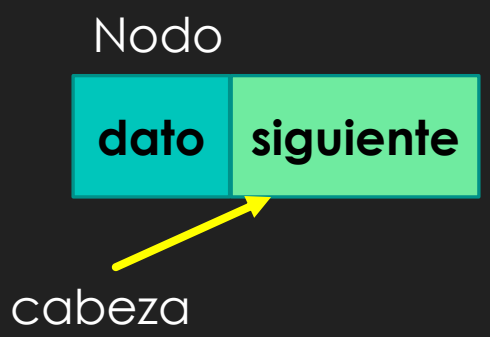



```
struct nodo
{
        int dato;
        Nodo * enlace;
};
```

1. Listas enlazadas

Punteros de cabeza y cola

- Los punteros de **cabeza** y **cola** son punteros de tipo **Nodo**
- Para indicar que la lista está vacía, los punteros pueden apuntar a **NULL**
- Puesto que es un puntero a estructura, para acceder a un miembro se usa el operador ->




```
struct nodo
{
        int dato;
        Nodo * enlace;
};

nodo * ptr_cabeza = NULL;
nodo * ptr_cola = NULL;
```

1. Listas enlazadas

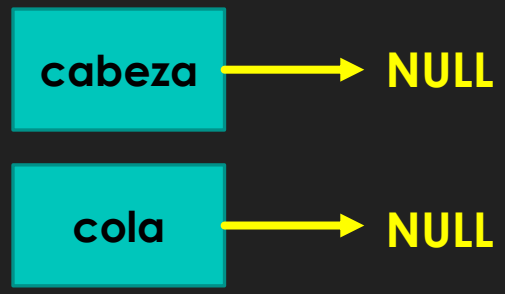
Pasos para la construcción de una lista enlazada (1)

- Para construir una lista enlazada se debe seguir el siguiente algoritmo:
  - **Paso 1**: declarar el tipo de dato y el puntero de cabeza
  - **Paso 2**: asignar memoria para un nuevo nodo
  - **Paso 3**: crear el primer elemento (cabeza) y los siguientes elementos de la lista
  - **Paso 4**: repetir hasta que no haya más elementos que crear

1. Listas enlazadas

Pasos para la construcción de una lista enlazada (2)

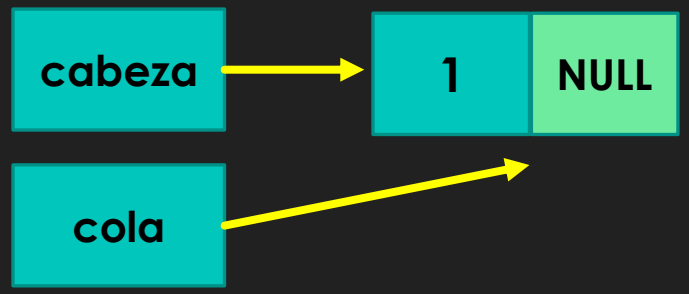
```
struct nodo
{
      int dato;
      Nodo * enlace;
};

nodo * ptr_cabeza = NULL;
nodo * ptr_cola = NULL;
```



1. Listas enlazadas

Pasos para la construcción de una lista enlazada (3)

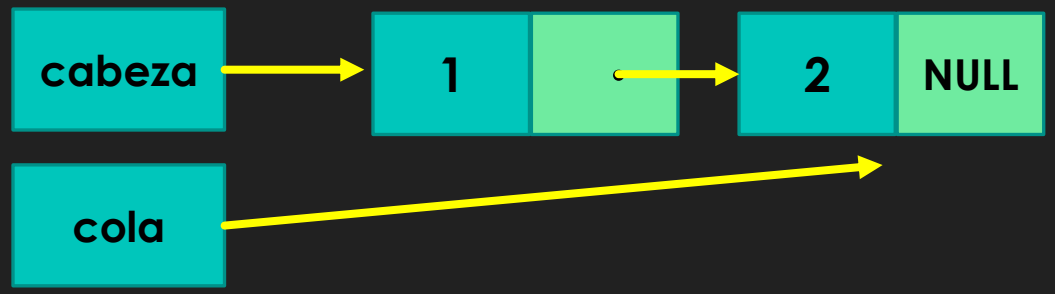
```
struct nodo
{
      int dato;
      Nodo * enlace;
};

nodo * ptr_cabeza = NULL;
nodo * ptr_cola = NULL;
```



1. Listas enlazadas

Pasos para la construcción de una lista enlazada (4)

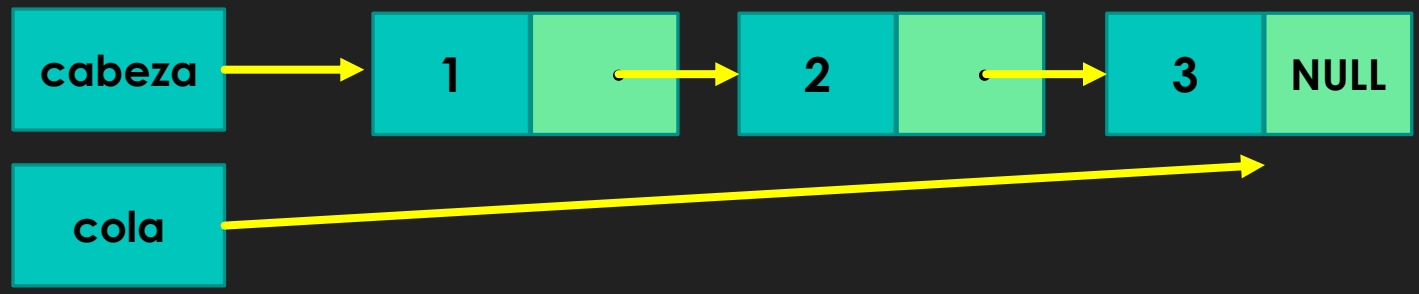
```
struct nodo
{
      int dato;
      Nodo * enlace;
};

nodo * ptr_cabeza = …;
nodo * ptr_cola = …;
```



1. Listas enlazadas

Pasos para la construcción de una lista enlazada (5)

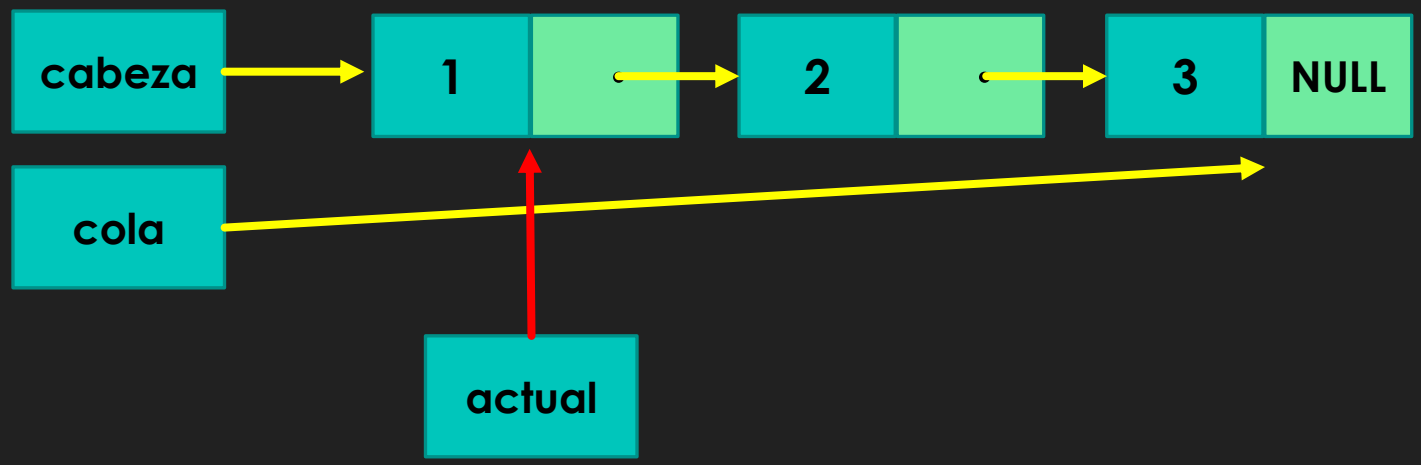
```
struct nodo
{
      int dato;
      Nodo * enlace;
};

nodo * ptr_cabeza = …;
nodo * ptr_cola = …;
```



1. Listas enlazadas

Pasos para la construcción de una lista enlazada (6)

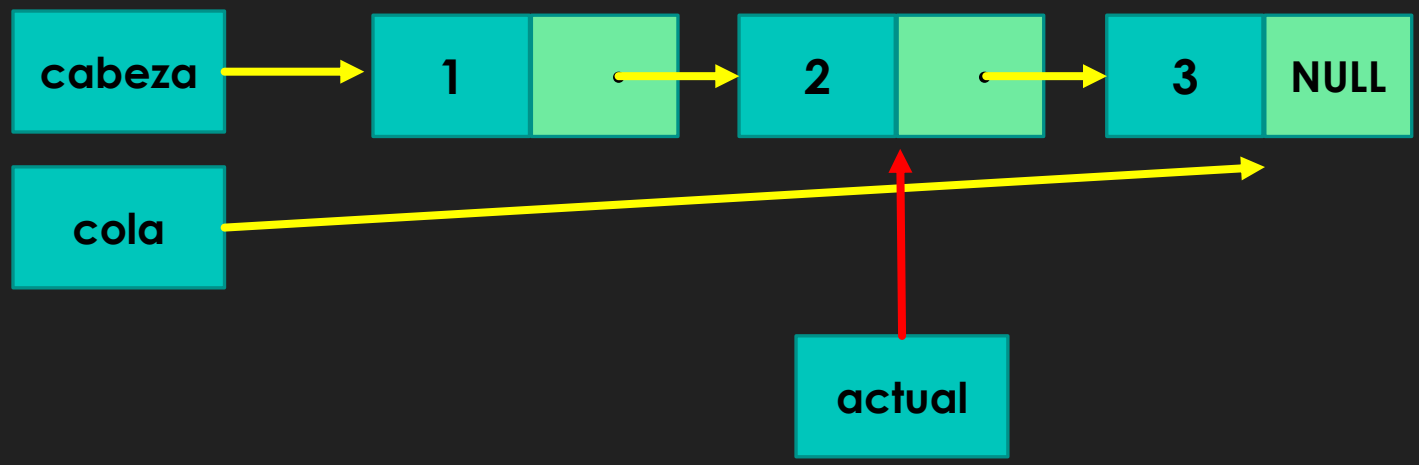
```
struct nodo
{
      int dato;
      Nodo * enlace;
};

nodo * ptr_cabeza = ...;
nodo * ptr_cola = ...;
actual * ptr_actual;
```



1. Listas enlazadas

Pasos para la construcción de una lista enlazada (7)

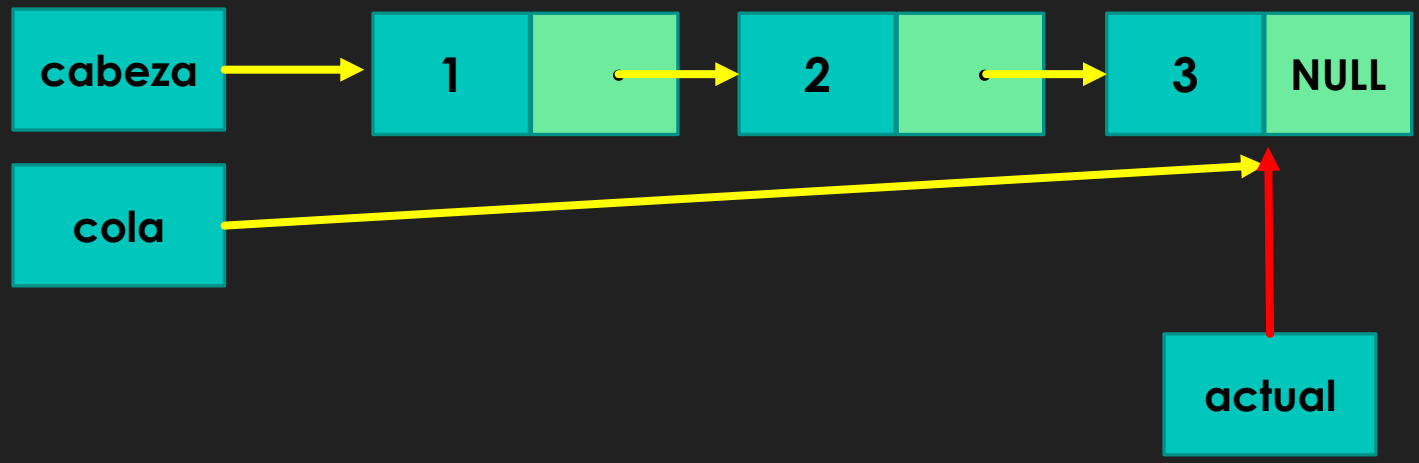
```
struct nodo
{
      int dato;
      Nodo * enlace;
};

nodo * ptr_cabeza = ...;
nodo * ptr_cola = ...;
actual * ptr_actual;
```



1. Listas enlazadas

Pasos para la construcción de una lista enlazada (8)

```
struct nodo
{
      int dato;
      Nodo * enlace;
};

nodo * ptr_cabeza = ...;
nodo * ptr_cola = ...;
actual * ptr_actual;
```



1. Listas enlazadas

Insertar un elemento en la lista (1)

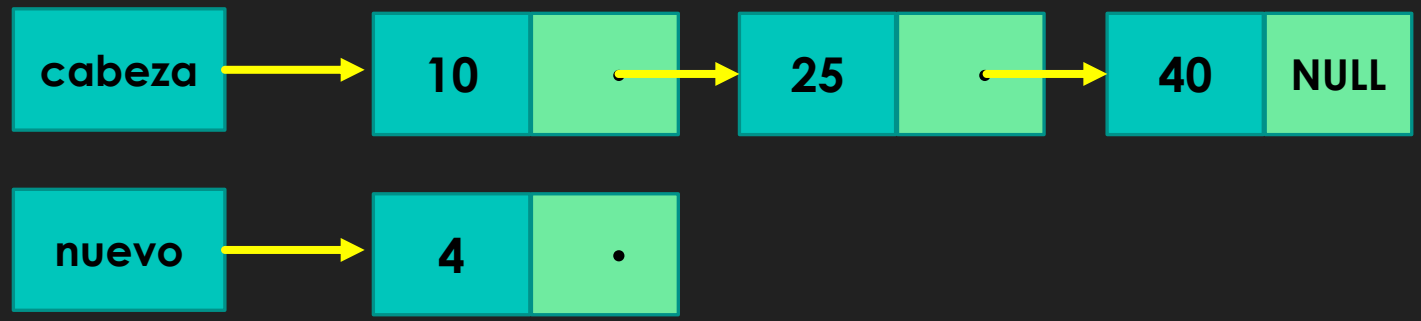
- Inserción en la cabeza de la lista:
  - **Paso 1**: crear un nuevo nodo
  - **Paso 2**: asignar los datos del nuevo nodo
  - **Paso 3**: enlazar el puntero del nuevo nodo a la cabeza de la lista
  - **Paso 4**: hacer que la cabeza de la lista enlace al nuevo nodo
- Este método también funciona cuando la lista está vacía y se quiere añadir el primer nodo

1. Listas enlazadas

Insertar un elemento en la lista (2)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 4 en la cabeza de la lista

Pasos 1-2

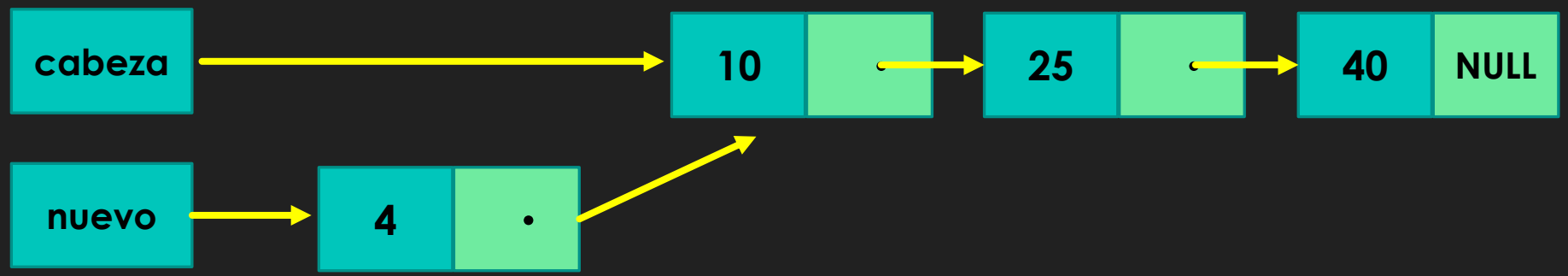


1. Listas enlazadas

Insertar un elemento en la lista (3)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 4 en la cabeza de la lista

Paso 3

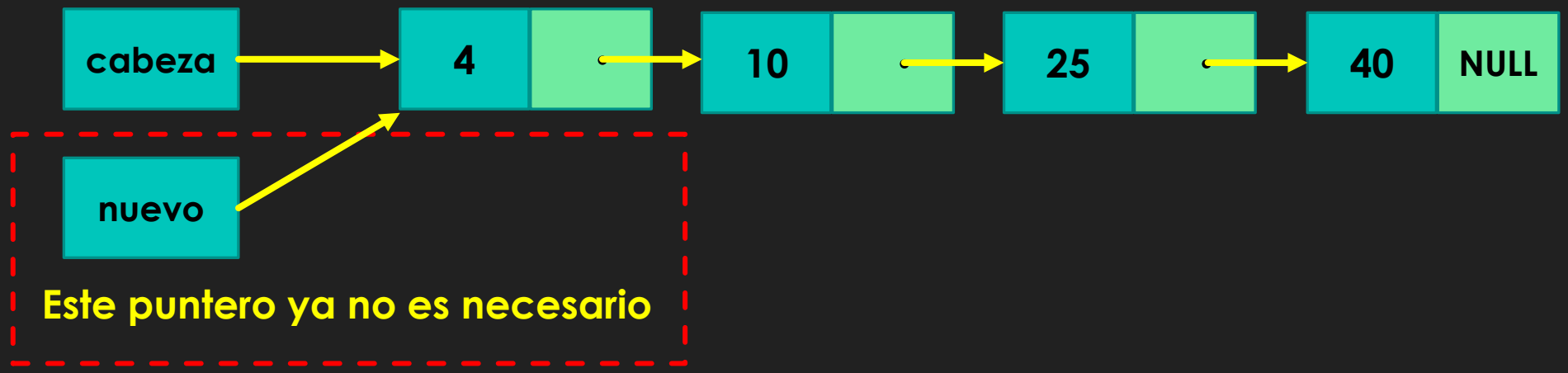


1. Listas enlazadas

Insertar un elemento en la lista (4)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 4 en la cabeza de la lista

Paso 4

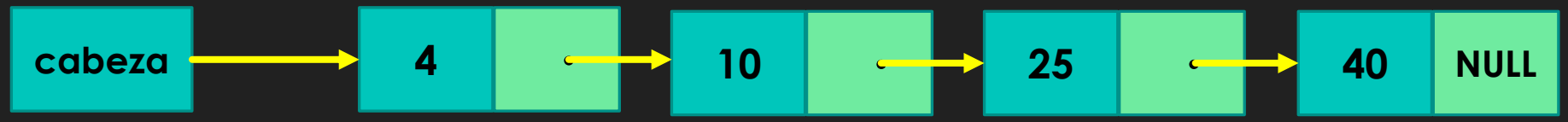


1. Listas enlazadas

Insertar un elemento en la lista (5)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 4 en la cabeza de la lista

Lista final



1. Listas enlazadas

Insertar un elemento en la lista (6)

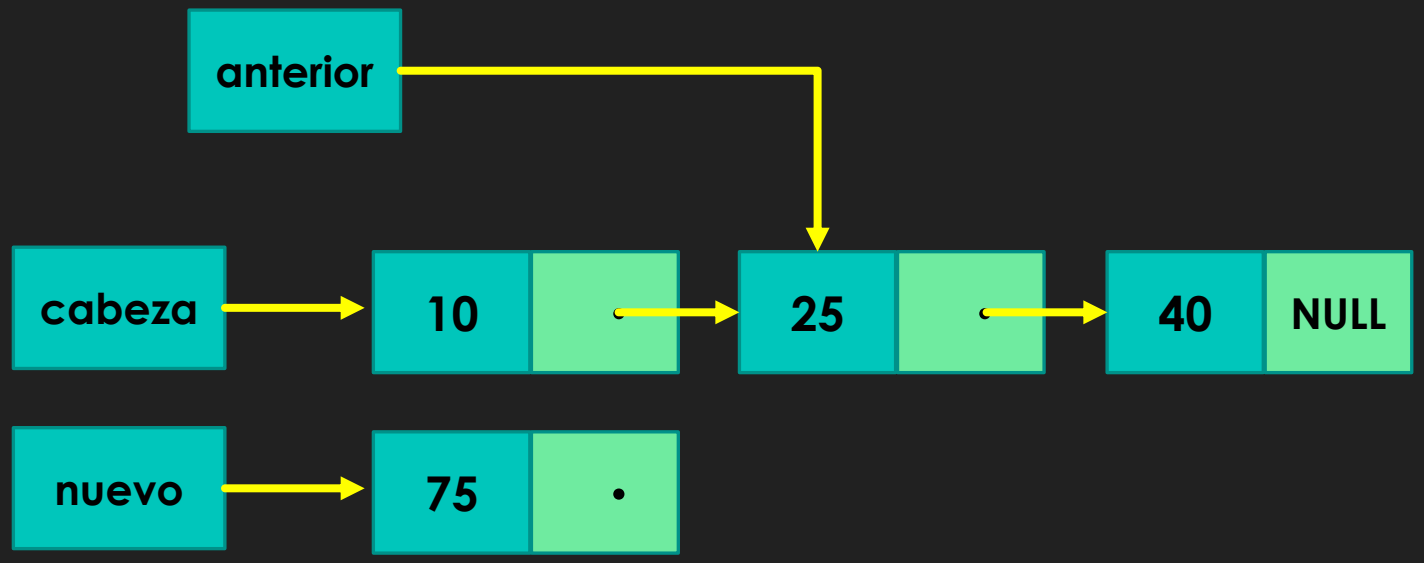
- Inserción en cualquier posición de la lista:
  - **Paso 1**: crear un nuevo nodo
  - **Paso 2**: asignar los datos del nuevo nodo
  - **Paso 3**: buscar la posición y enlazar el puntero del nuevo nodo al nodo que va después
  - **Paso 4**: crear un puntero al nodo que va antes y enlazarlo con el nuevo nodo que hemos creado

1. Listas enlazadas

Insertar un elemento en la lista (7)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 75 después del elemento 25

Pasos 1-2

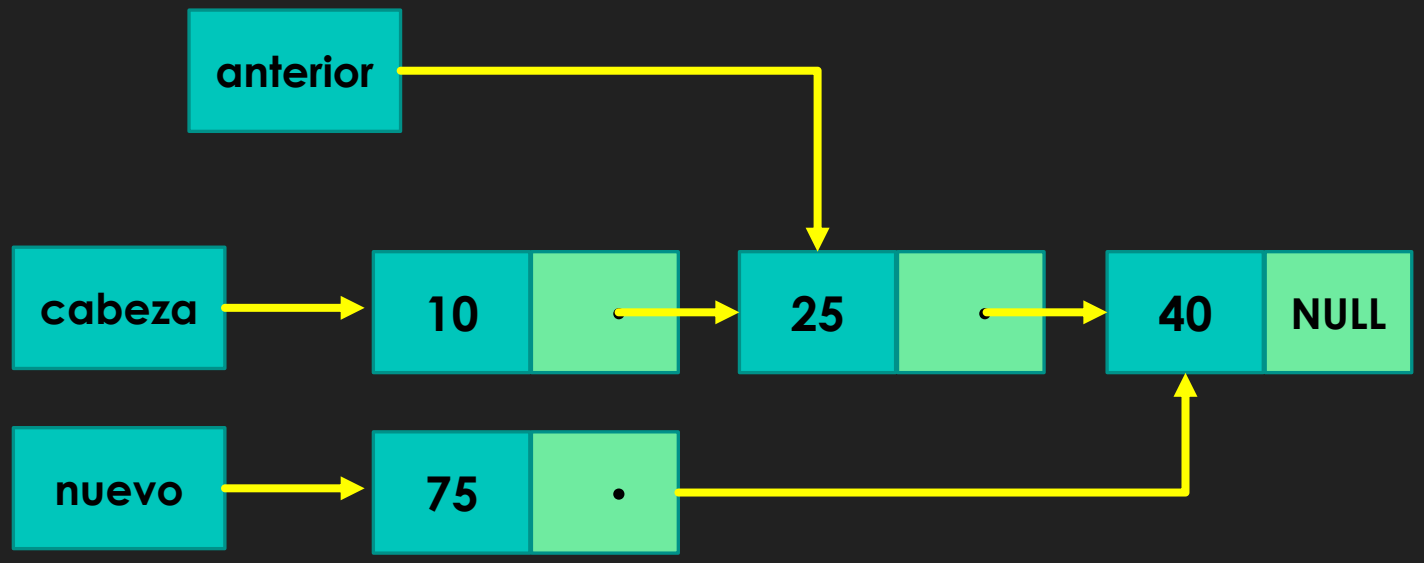


1. Listas enlazadas

Insertar un elemento en la lista (8)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 75 después del elemento 25

Paso 3

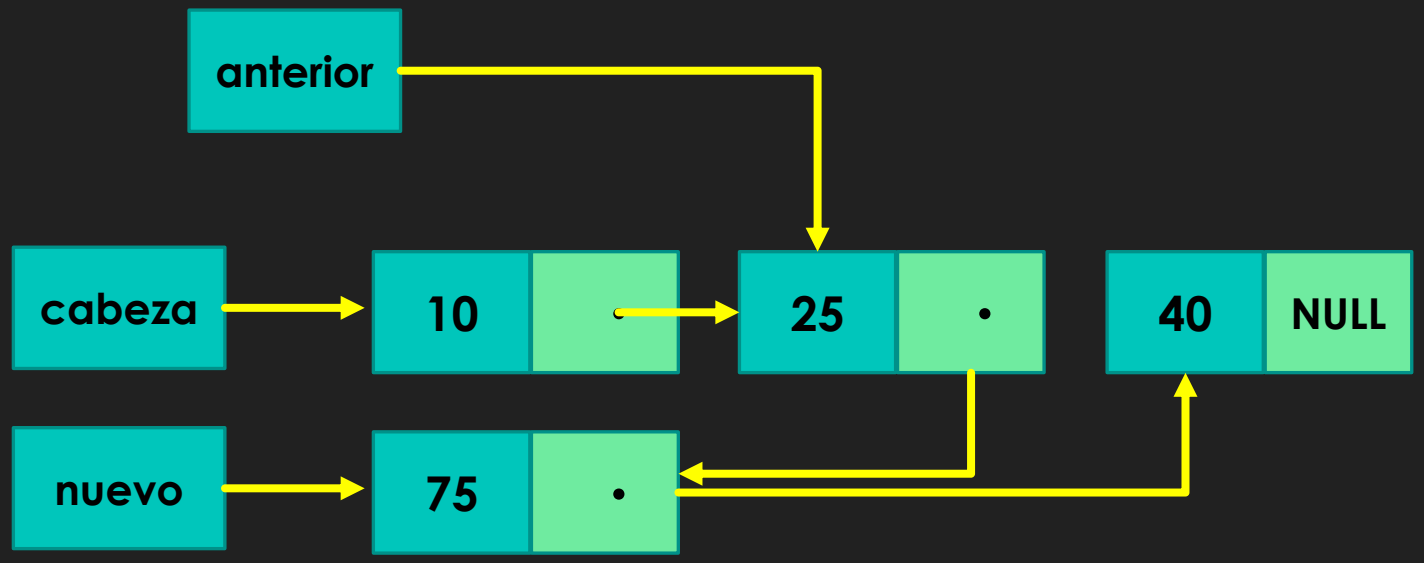


1. Listas enlazadas

Insertar un elemento en la lista (9)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 75 después del elemento 25

Paso 4

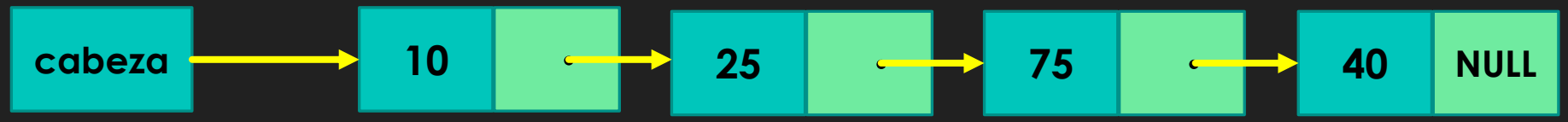


1. Listas enlazadas

Insertar un elemento en la lista (10)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 75 después del elemento 25

Lista final



1. Listas enlazadas

Insertar un elemento en la lista (11)

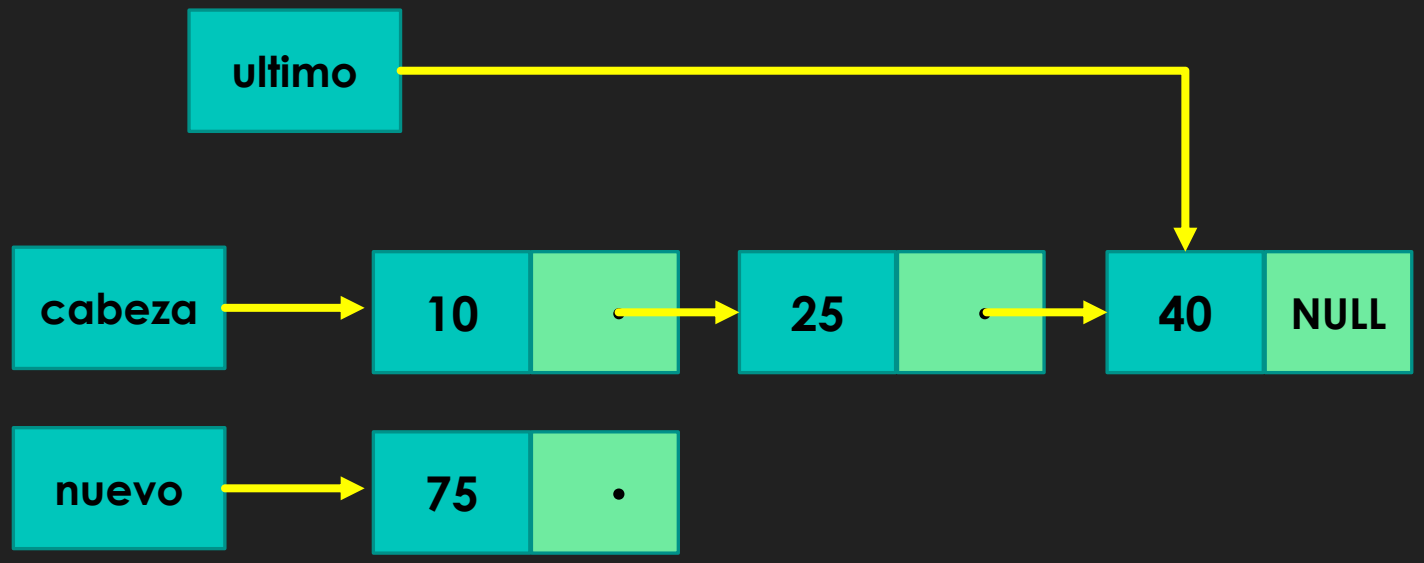
- Inserción al final de la lista:
  - **Paso 1**: crear un nuevo nodo
  - **Paso 2**: asignar los datos del nuevo nodo
  - **Paso 3**: buscar el último elemento de la lista (o utilizar un puntero al último elemento, si lo tenemos)
  - **Paso 4**: modificar el puntero de último nodo de la lista para que apunte al nuevo
  - **Paso 5**: modificar el puntero del nuevo nodo para que apunte a **NULL**

1. Listas enlazadas

Insertar un elemento en la lista (12)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 75 al final

Pasos 1-2-3

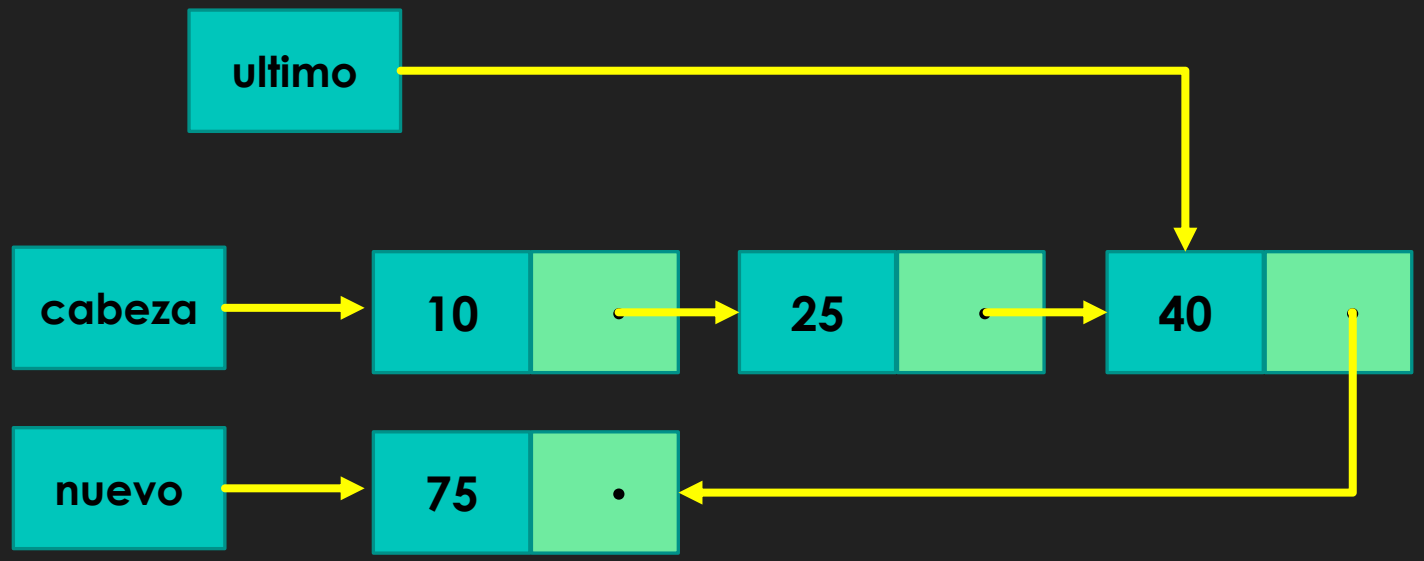


1. Listas enlazadas

Insertar un elemento en la lista (13)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 75 al final

Paso 4

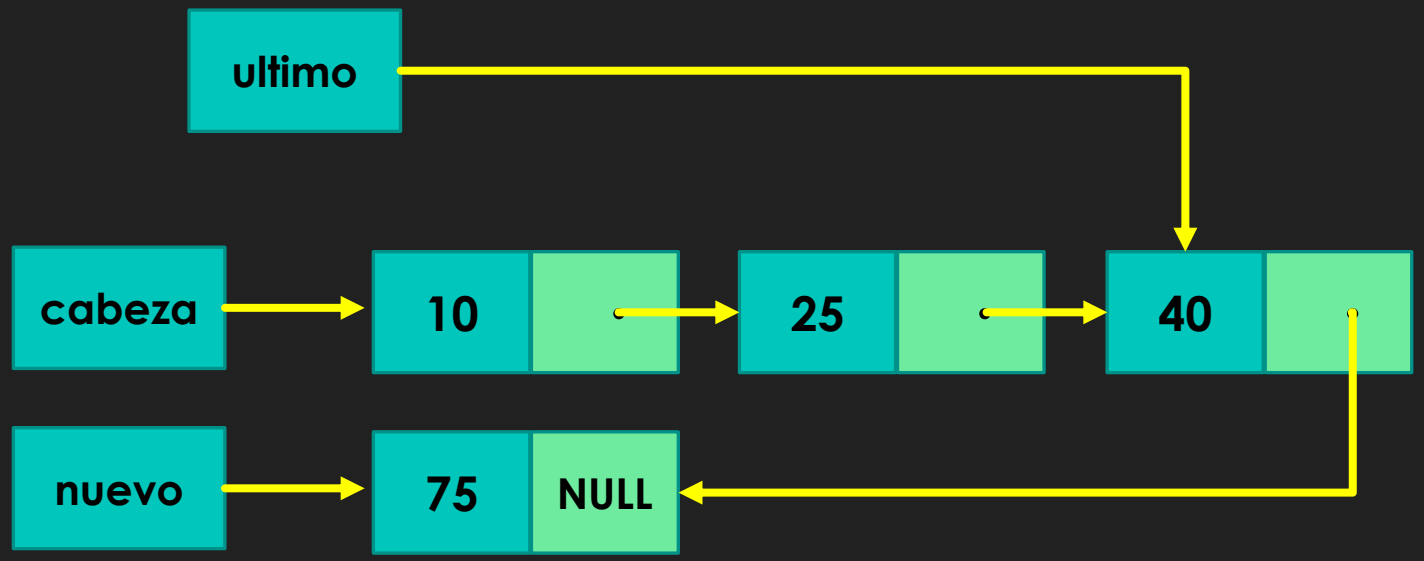


1. Listas enlazadas

Insertar un elemento en la lista (14)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 75 al final

Paso 5

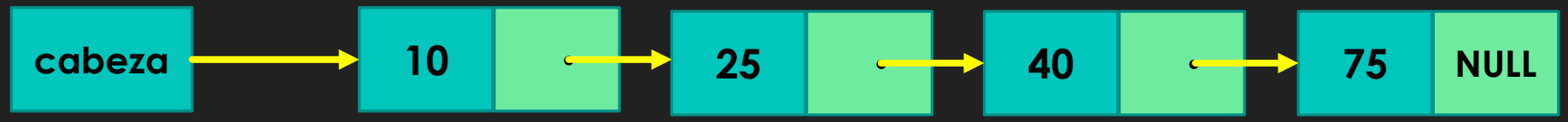


1. Listas enlazadas

Insertar un elemento en la lista (14)

- Sea una lista que contiene los elementos 10, 25 y 40
- Queremos insertar el elemento 75 al final

Lista final



1. Listas enlazadas

Buscar un elemento de la lista

- Búsqueda de un elemento dentro de la lista:
  - **Paso 1**: recorrer la lista completa, nodo a nodo
  - **Paso 2**: si se encuentra el nodo buscado, devolver un puntero a dicho nodo
  - **Paso 3**: si no se encuentra el nodo buscado, devolver **NULL**

1. Listas enlazadas

Eliminar un elemento de la lista (1)

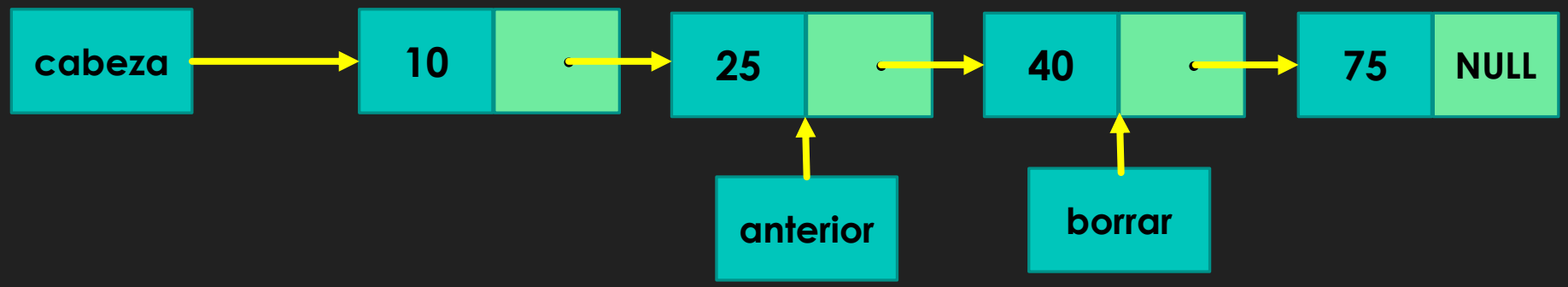
- Eliminación de un elemento de la lista:
  - **Paso 1**: buscar el nodo que se quiere borrar, crear un puntero a dicho nodo y al nodo anterior
  - **Paso 2**: enlazar el puntero del nodo anterior al nodo siguiente al nodo que se quiere borrar
  - **Paso 3**: si el nodo que se va a borrar es el primero, modificar la cabeza de lista para apuntar al siguiente
  - **Paso 4**: liberar la memoria por el nodo que se quiere borrar

1. Listas enlazadas

Eliminar un elemento de la lista (2)

- Sea una lista que contiene los elementos 10, 25, 40 y 75
- Queremos borrar el elemento 40 (no vamos a borrar el primero; el paso 3 es innecesario)

Paso 1

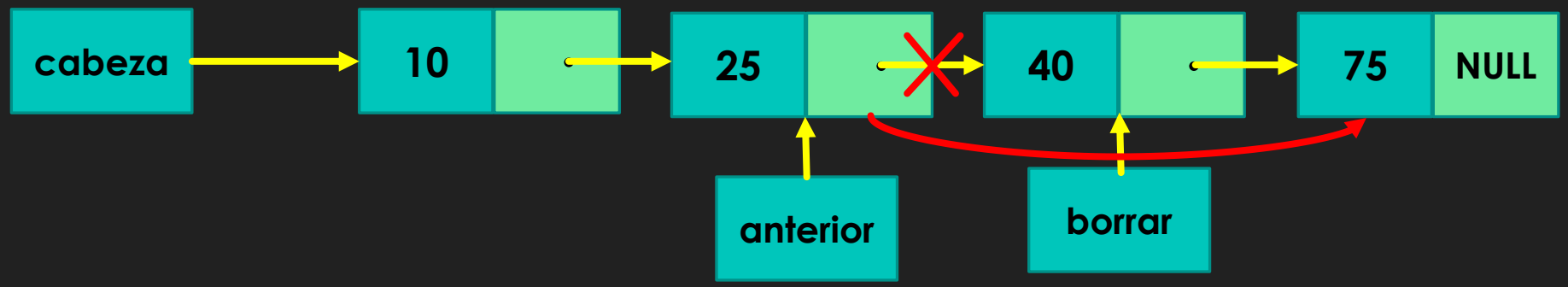


1. Listas enlazadas

Eliminar un elemento de la lista (3)

- Sea una lista que contiene los elementos 10, 25, 40 y 75
- Queremos borrar el elemento 40 (no vamos a borrar el primero; el paso 3 es innecesario)

Paso 2

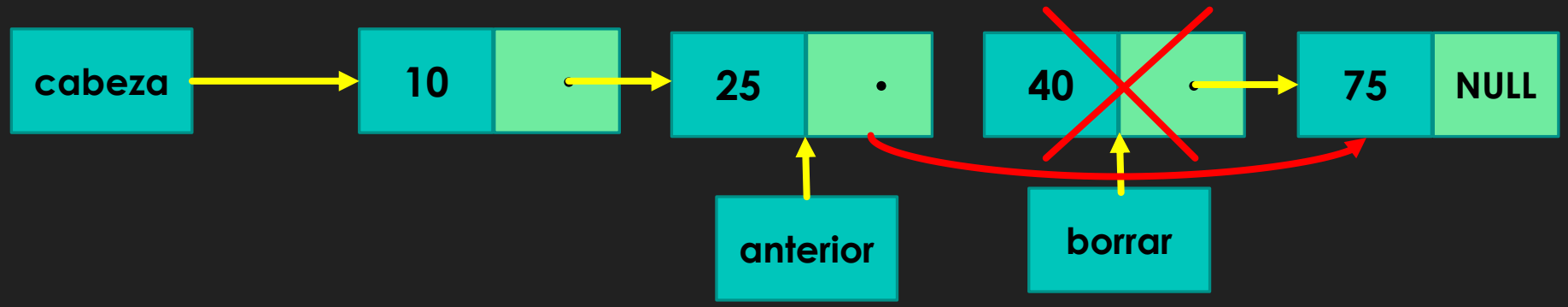


1. Listas enlazadas

Eliminar un elemento de la lista (4)

- Sea una lista que contiene los elementos 10, 25, 40 y 75
- Queremos borrar el elemento 40 (no vamos a borrar el primero; el paso 3 es innecesario)

Paso 4

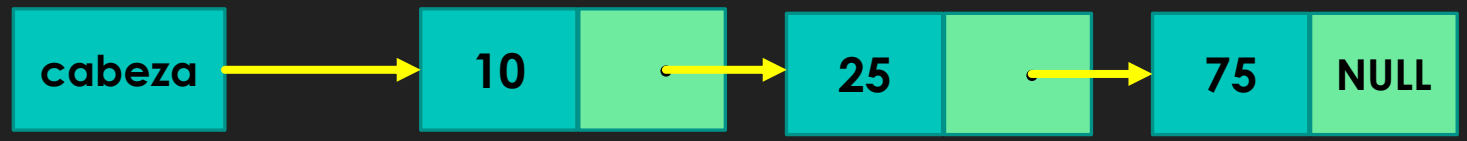


1. Listas enlazadas

Eliminar un elemento de la lista (5)

- Sea una lista que contiene los elementos 10, 25, 40 y 75
- Queremos borrar el elemento 40 (no vamos a borrar el primero; el paso 3 es innecesario)

Lista final

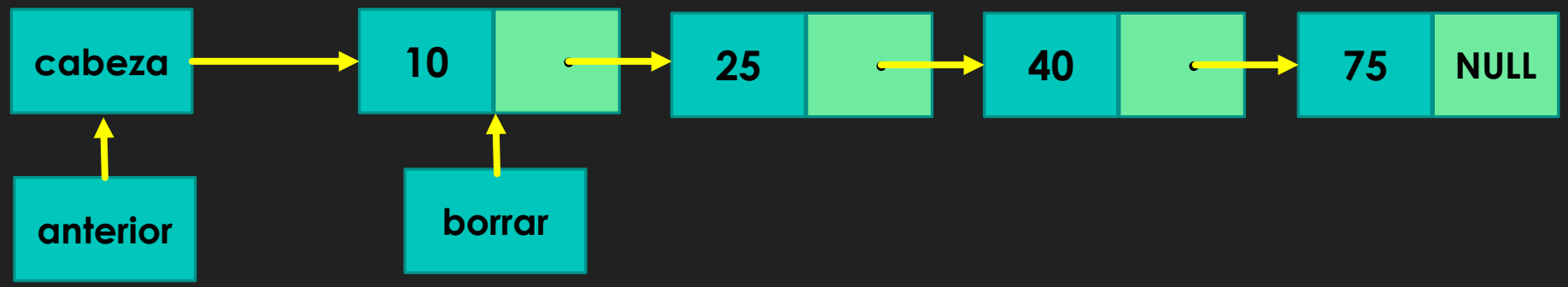


1. Listas enlazadas

Eliminar un elemento de la lista (6)

- Sea una lista que contiene los elementos 10, 25, 40 y 75
- Queremos borrar el elemento 10 (primero de la lista)

Paso 1

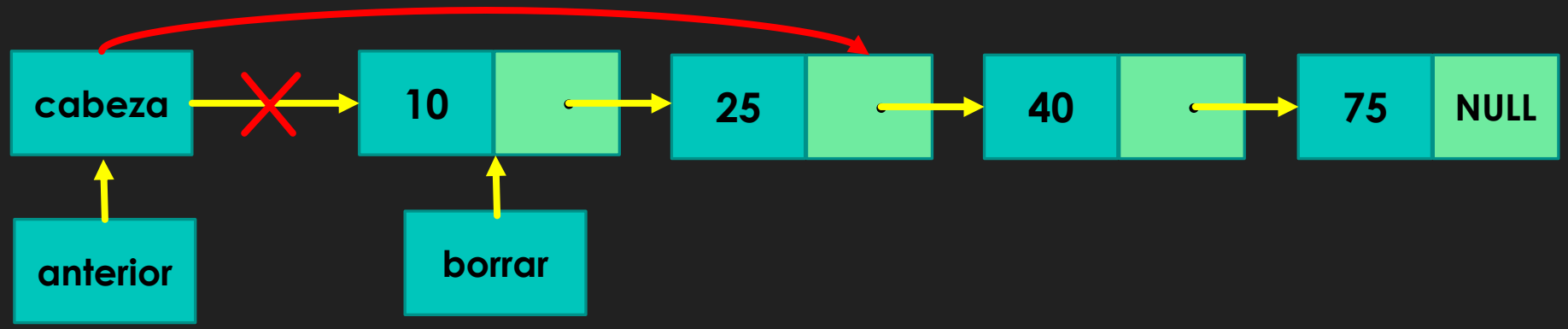


1. Listas enlazadas

Eliminar un elemento de la lista (7)

- Sea una lista que contiene los elementos 10, 25, 40 y 75
- Queremos borrar el elemento 10 (primero de la lista

Paso 3

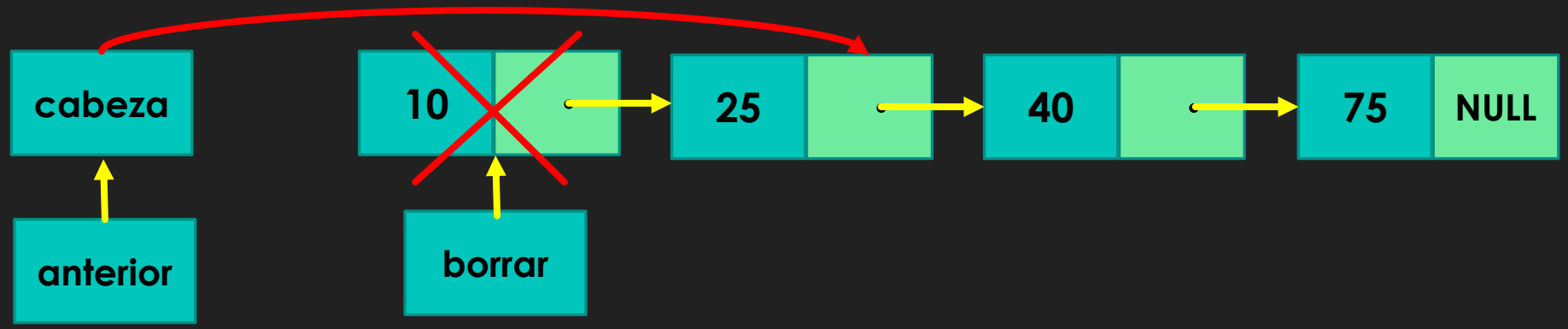


1. Listas enlazadas

Eliminar un elemento de la lista (8)

- Sea una lista que contiene los elementos 10, 25, 40 y 75
- Queremos borrar el elemento 10 (primero de la lista)

Paso 3

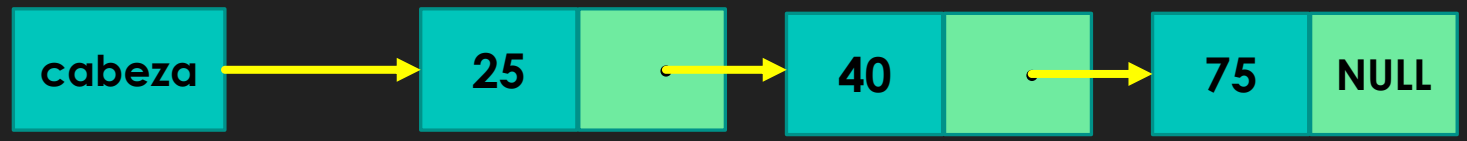


1. Listas enlazadas

Eliminar un elemento de la lista (9)

- Sea una lista que contiene los elementos 10, 25, 40 y 75
- Queremos borrar el elemento 10 (primero de la lista)

Lista final



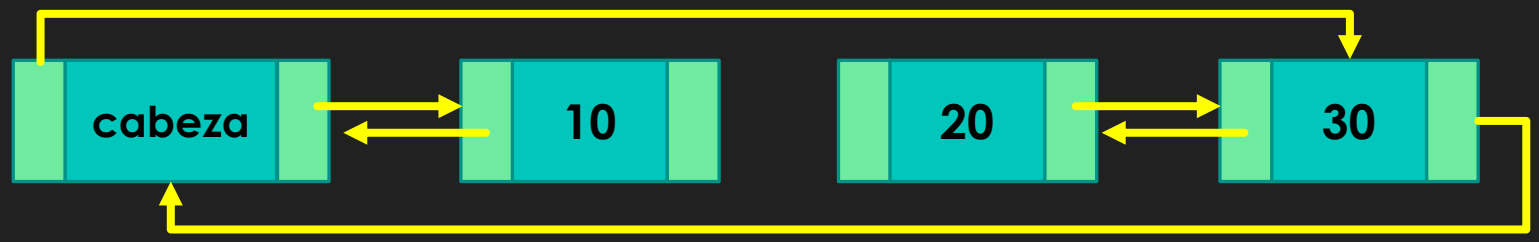
1. Listas enlazadas

Listas doblemente enlazadas

- Permiten acceder a los elementos en cualquier orden (izquierda o derecha)
- Los nodos tienen dos punteros, en lugar de uno

Ejemplo

```
struct nodo
{
        int dato;
        Nodo * izquierda;
        Nodo * derecha;
};
```

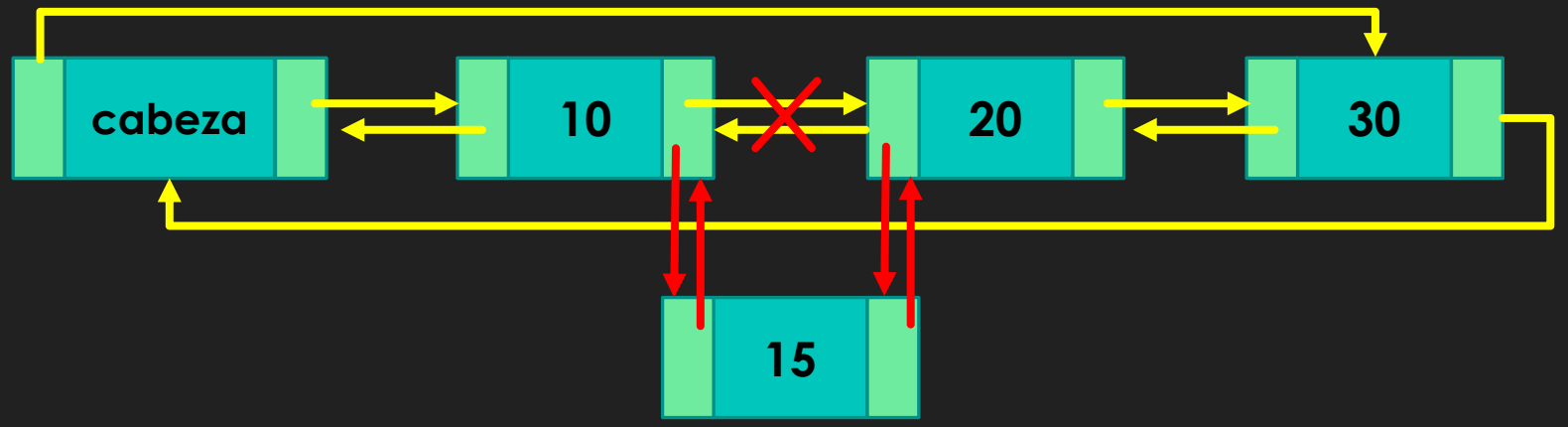


1. Listas enlazadas

Inserción en listas doblemente enlazadas (1)

- Además de al principio o al final, podemos insertar a la izquierda o a la derecha de un nodo

Ejemplo

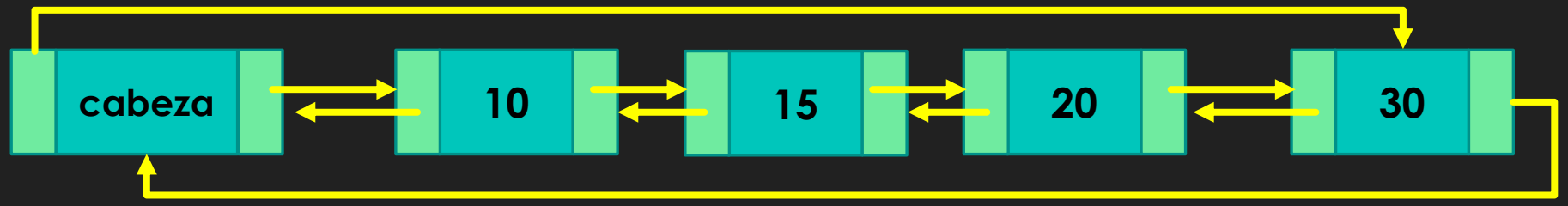


1. Listas enlazadas

Inserción en listas doblemente enlazadas (2)

- Además de al principio o al final, podemos insertar a la izquierda o a la derecha de un nodo

Ejemplo

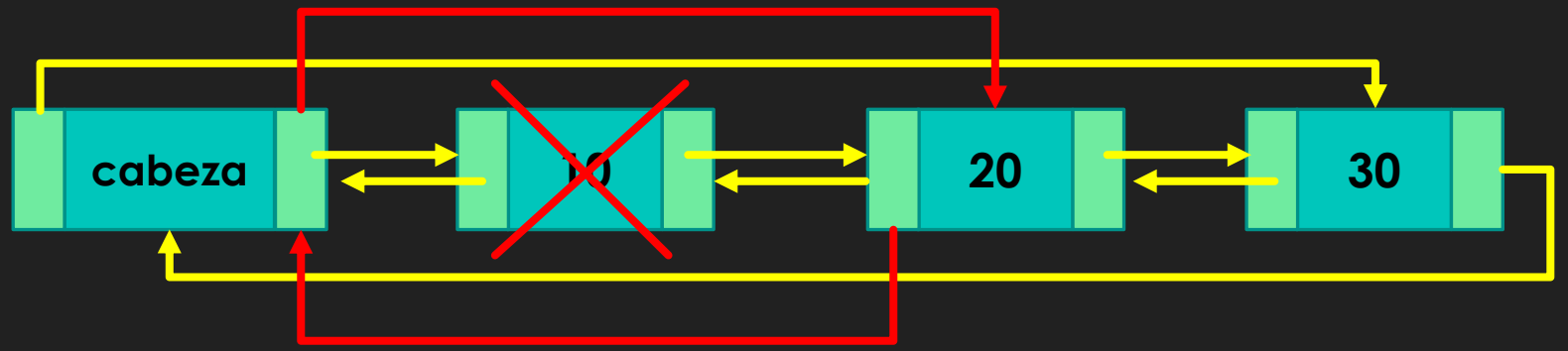


1. Listas enlazadas

Eliminación en listas doblemente enlazadas (1)

- Además de al principio o al final, podemos borrar a la izquierda o a la derecha de un nodo

Ejemplo

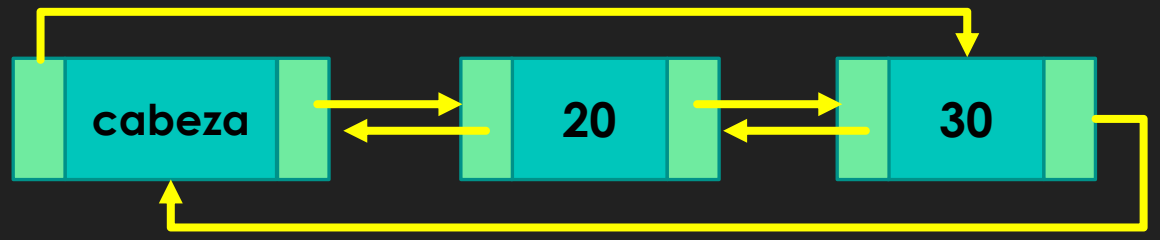


1. Listas enlazadas

Eliminación en listas doblemente enlazadas (2)

- Además de al principio o al final, podemos borrar a la izquierda o a la derecha de un nodo

Ejemplo

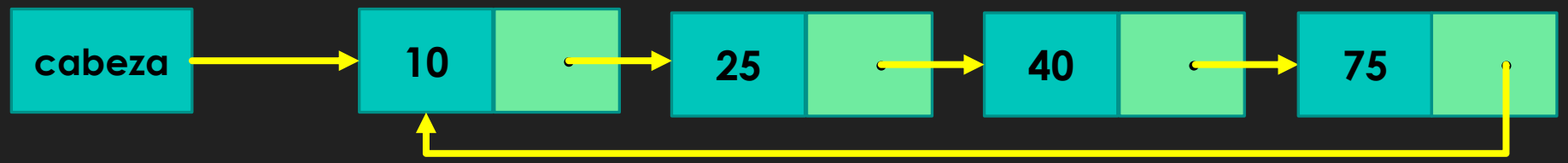


1. Listas enlazadas

Listas circulares

- Las listas circulares no tienen ni principio ni fin
- Sin embargo, resulta útil tener un puntero para delimitar el comienzo de la lista
- En las operaciones de la lista hay que tener en cuenta que el último nodo apunta al primero

Ejemplo



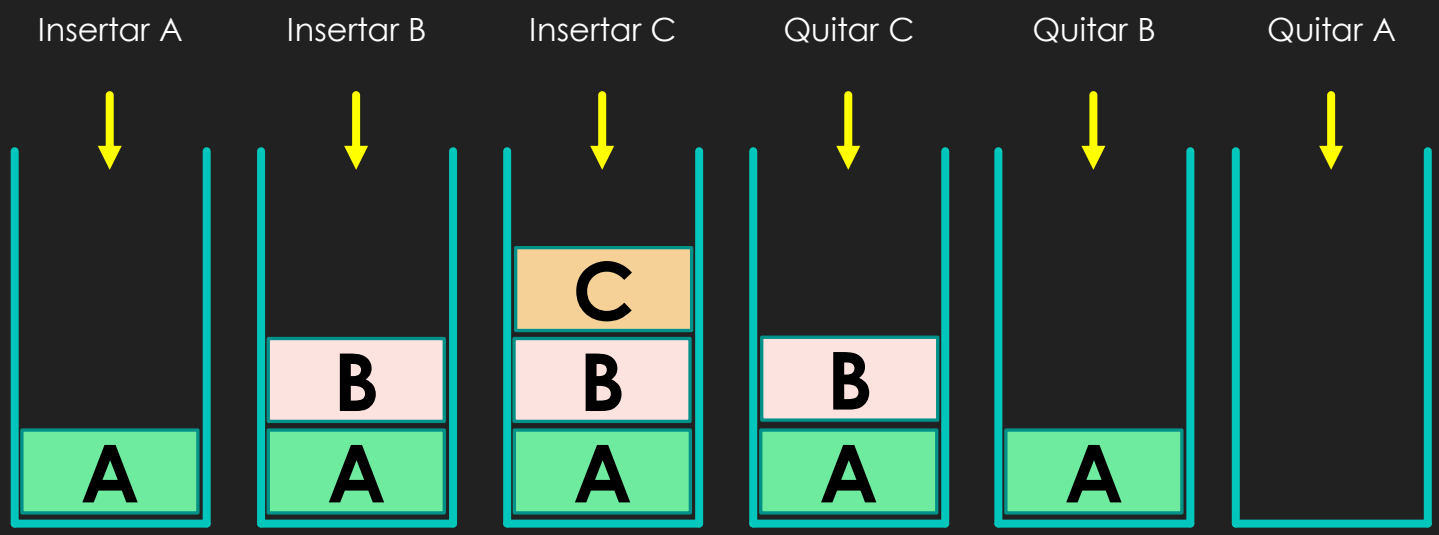
2. Pilas

Concepto de pila (1)

- Una **pila** (**stack**) es una colección de elementos a los que se accede por un único extremo
- En la pila se añaden o borran elementos a través de su **parte superior** (**cima**)
- Debido a esta propiedad, la pila es una estructura **LIFO** (**last-in**, **first-out**)
- Las operaciones usuales son:
  - **Insertar** (**push**): añade un elemento en la cima de la pila
  - **Quitar** (**pop**): quita el elemento de la cima de la pila
  - **Vacía**: comprueba si la pila está vacía (cima = NULL)
  - **Tamaño**: consiste en contar el número de elementos (navegar desde la cima hasta NULL)

2. Pilas

Concepto de pila (2)

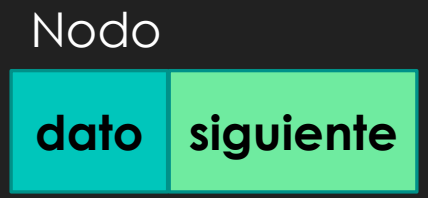


Entrada: A B C
Salida: C B A

2. Pilas

Concepto de pila (3)

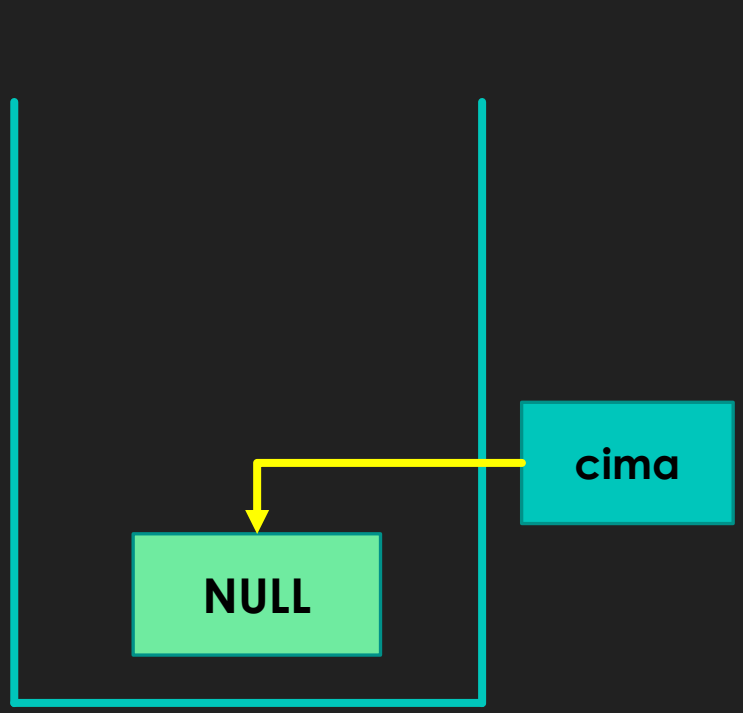
- Podemos reutilizar la misma estructura de nodo
- En este caso solo necesitaremos un **puntero de posición**, hacia la **cima** de la pila

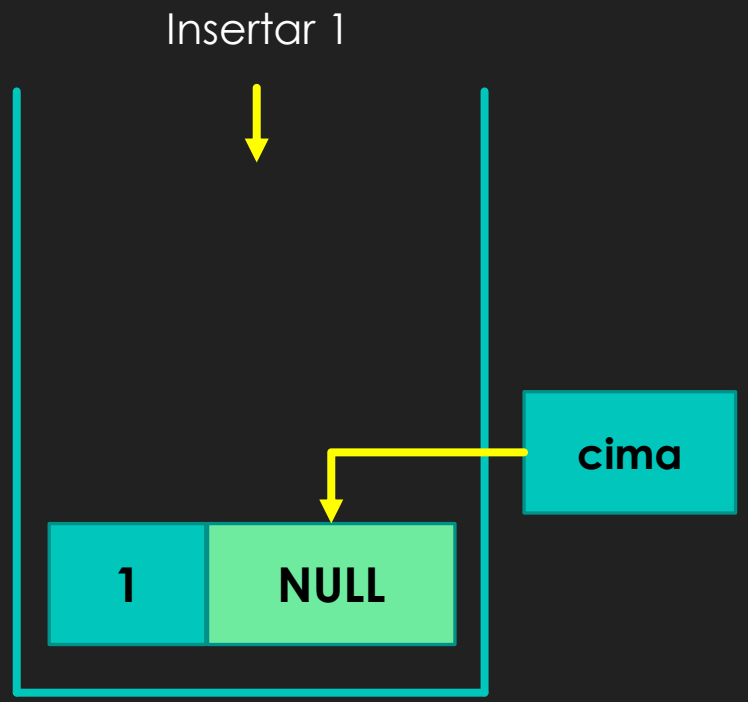



```
struct nodo
{
        int dato;
        Nodo * enlace;
};

nodo * ptr_cima = NULL;
```

2. Pilas

Insertar elementos (1)

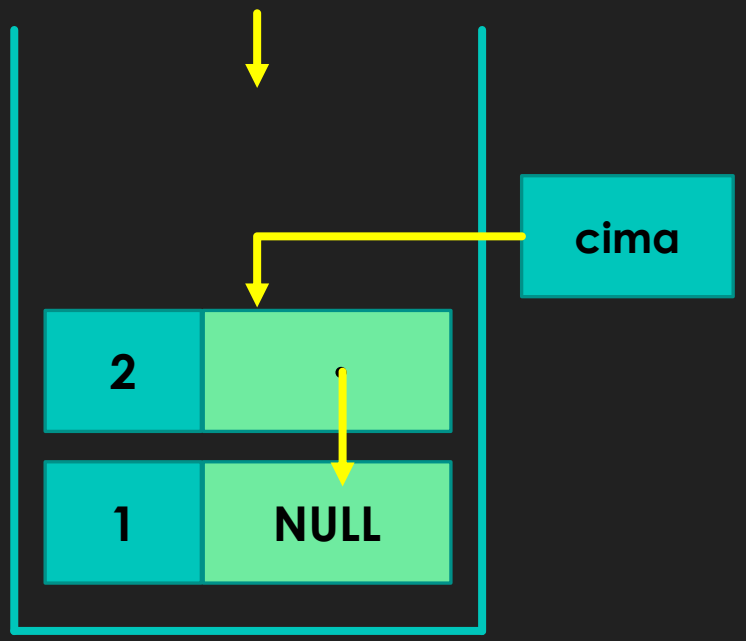




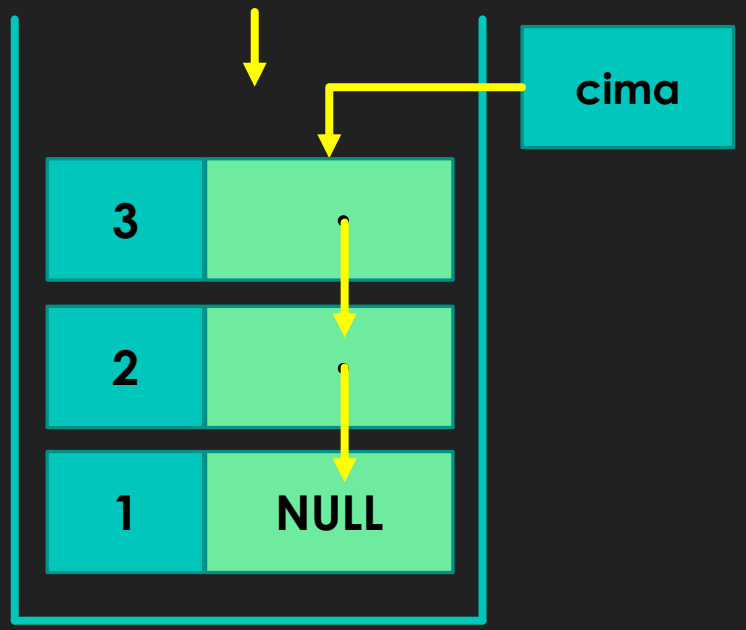
2. Pilas

Insertar elementos (2)

Insertar 2



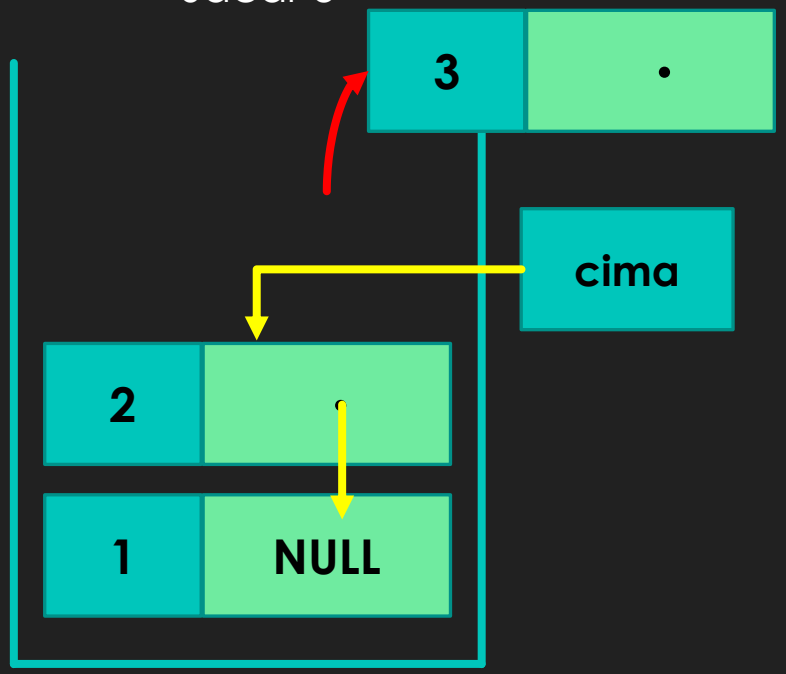
Insertar 3



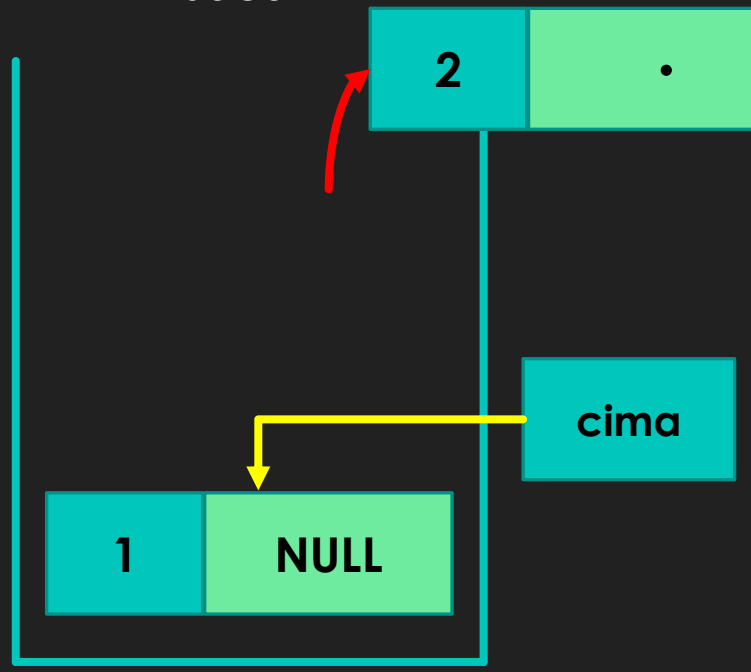
2. Pilas

Sacar elementos

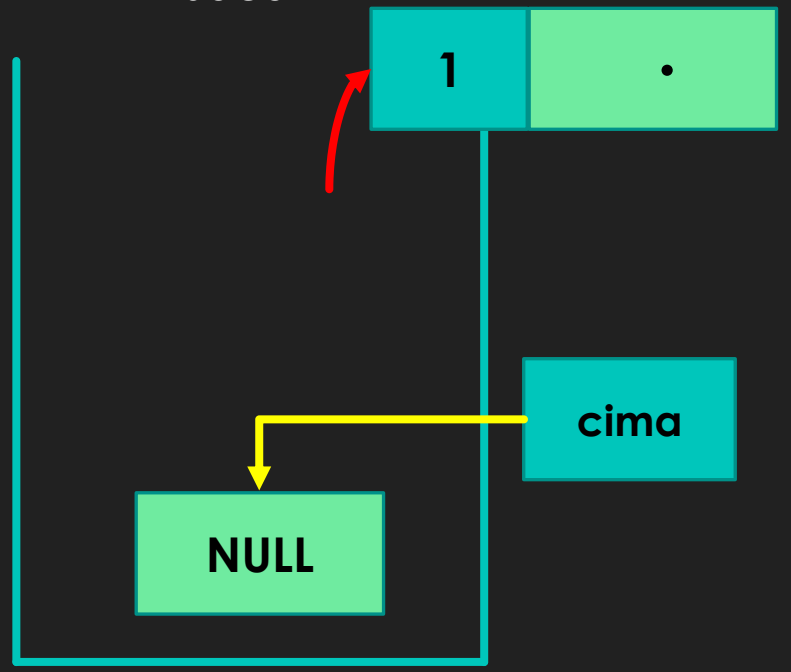
Sacar 3



Sacar 2



Sacar 1



3. Colas

Concepto de cola (1)

- Una **cola** es una estructura de datos que almacena elementos en una lista
- Un elemento se inserta en la cola en la **parte final** y se borra por la **parte inicial**
- Debido a esta propiedad, la cola es una estructura **FIFO** (**first-in**, **first-out**)
- Las operaciones usuales son:
  - **Insertar**: añade un elemento al final de la cola
  - **Quitar**: quita el elemento del principio de la cola
  - **Vacía**: comprueba si la cola está vacía

3. Colas

Concepto de cola (2)

- Podemos reutilizar la misma estructura de nodo
- En este caso usaremos dos **puntero de posición**, para el **frente** y el **fin** de la cola

Nodo

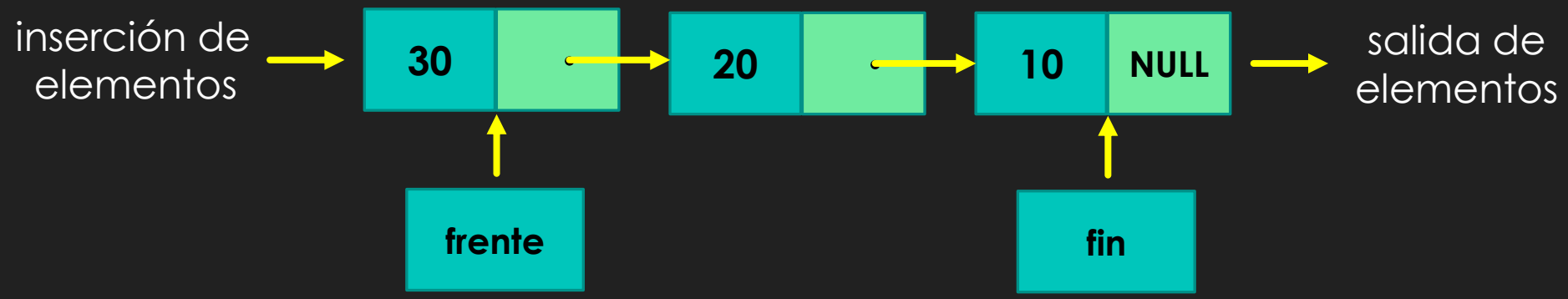

```
struct nodo
{
        int dato;
        Nodo * enlace;
};

nodo * ptr_frente = NULL;
nodo * ptr_fin = NULL;
```

3. Colas

Concepto de cola (3)

- Sea una cola que contiene los elementos 10, 20 y 30
- Si la cola está vacía, al inicio **frente** y **fin** apuntarán a **NULL**

Ejemplo

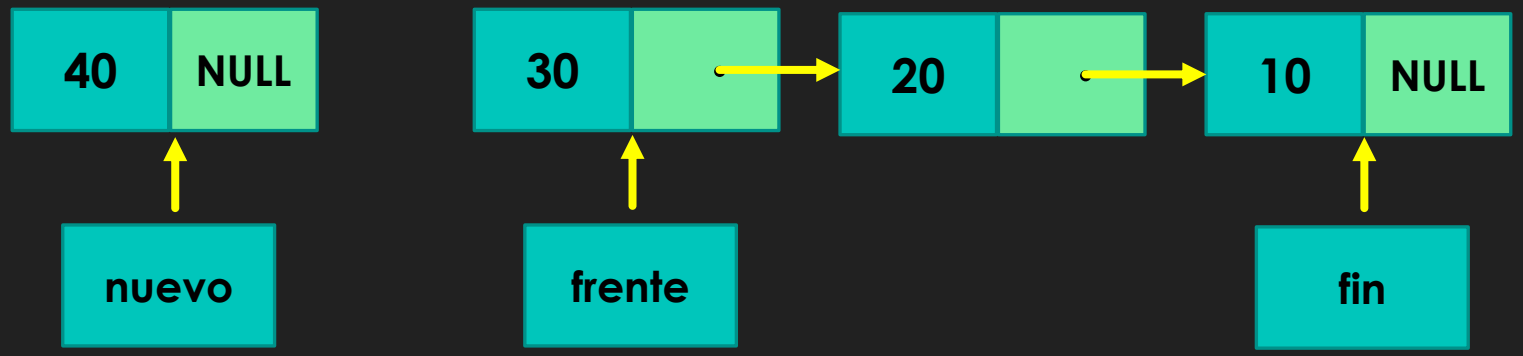


3. Colas

Insertar elementos (1)

- Sea una cola que contiene los elementos 10, 20 y 30
- Queremos insertar el elemento 40

Paso 1: crear el nuevo nodo

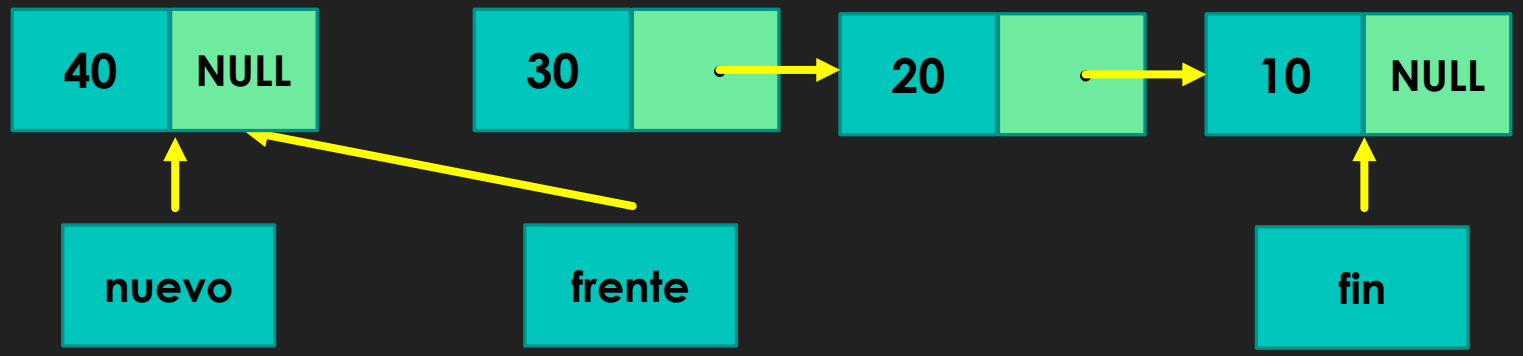


3. Colas

Insertar elementos (2)

- Sea una cola que contiene los elementos 10, 20 y 30
- Queremos insertar el elemento 40

Paso 2: modificar el puntero frente

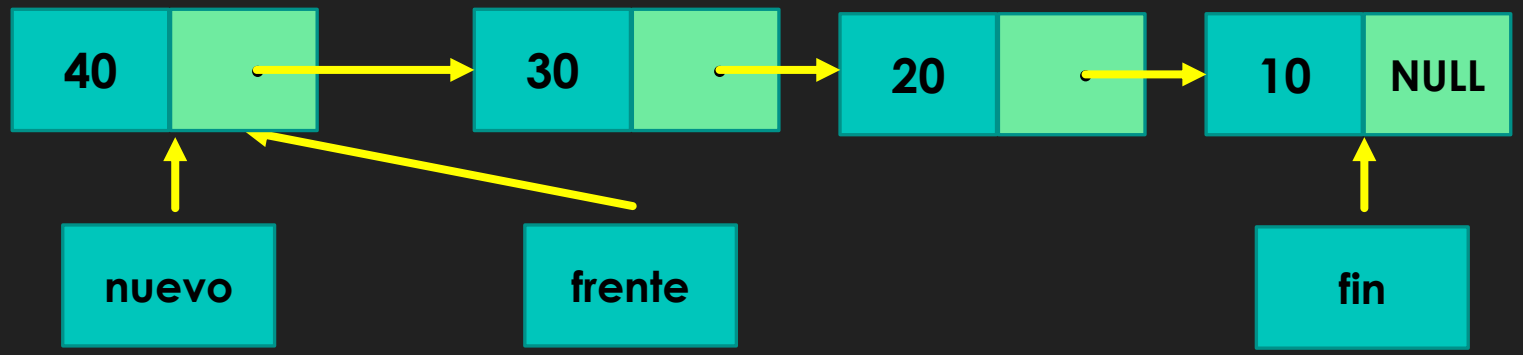


3. Colas

Insertar elementos (3)

- Sea una cola que contiene los elementos 10, 20 y 30
- Queremos insertar el elemento 40

Paso 3: actualizar el puntero siguiente del nuevo nodo

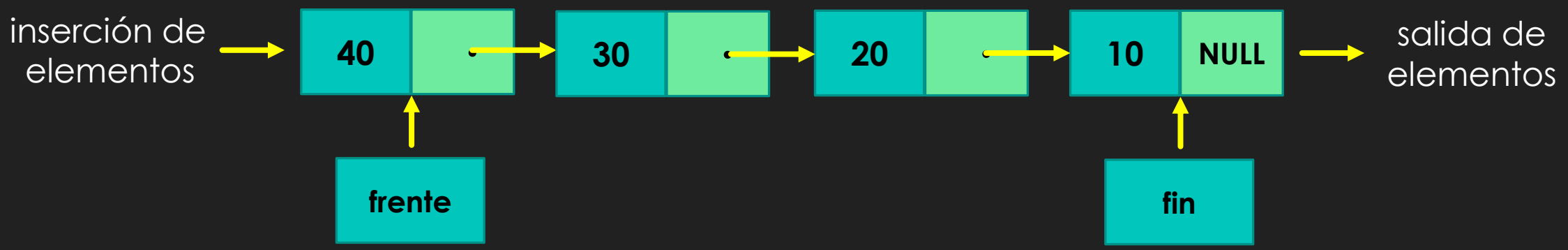


3. Colas

Insertar elementos (4)

- Sea una cola que contiene los elementos 10, 20 y 30
- Queremos insertar el elemento 40

Cola final

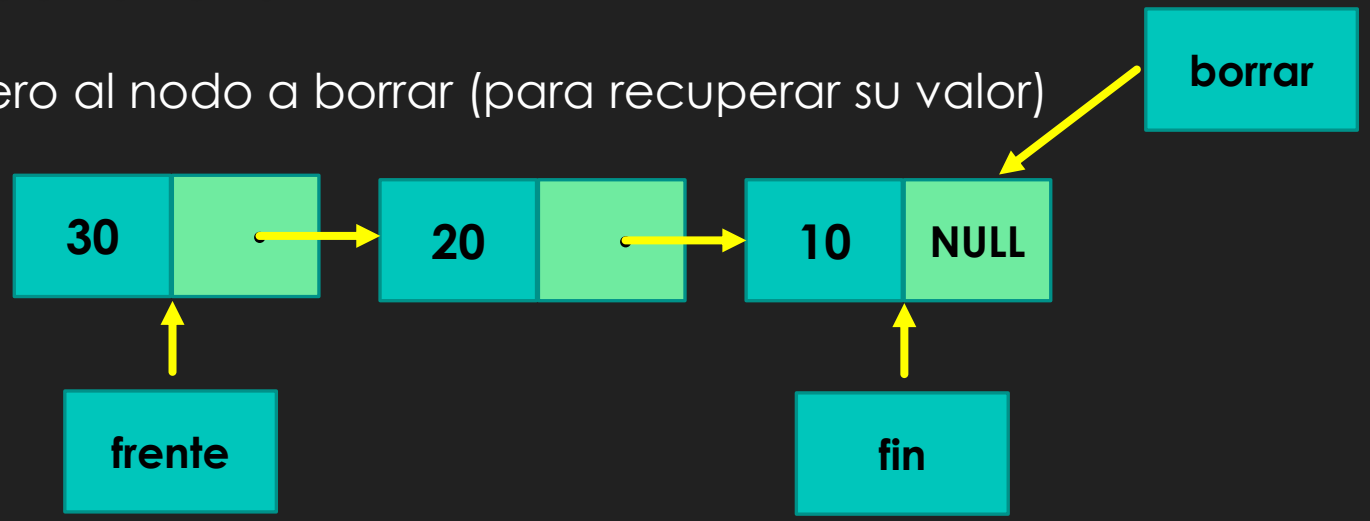


3. Colas

Borrar elementos (1)

- Sea una cola que contiene los elementos 10, 20 y 30
- Queremos borrar el elemento 10

Paso 1: crear un puntero al nodo a borrar (para recuperar su valor)

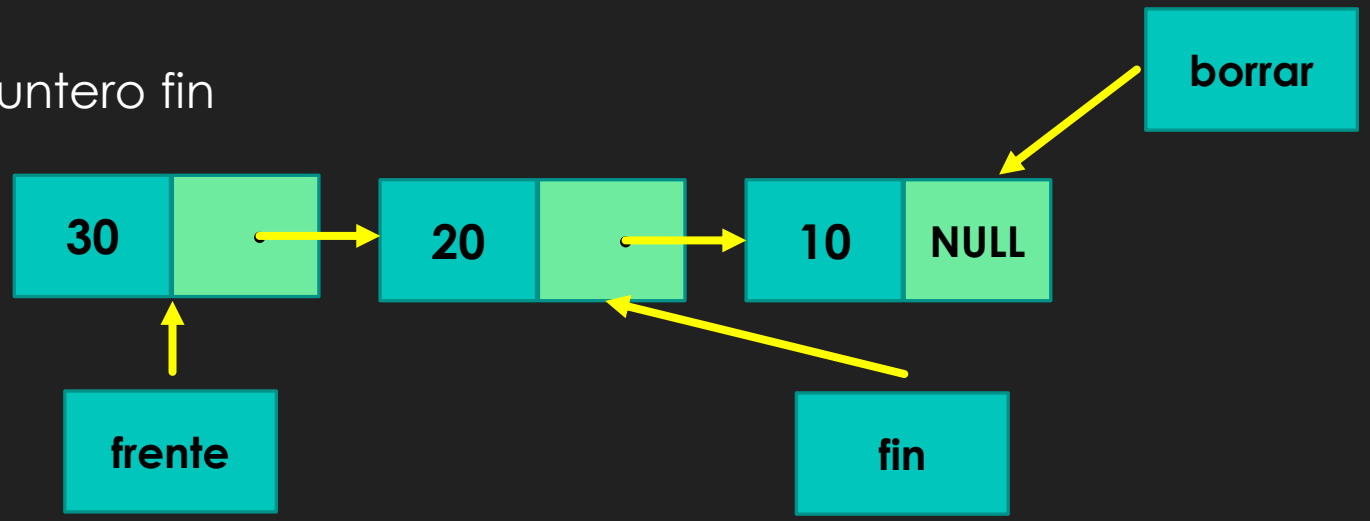


3. Colas

Borrar elementos (2)

- Sea una cola que contiene los elementos 10, 20 y 30
- Queremos borrar el elemento 10

Paso 2: actualizar el puntero fin

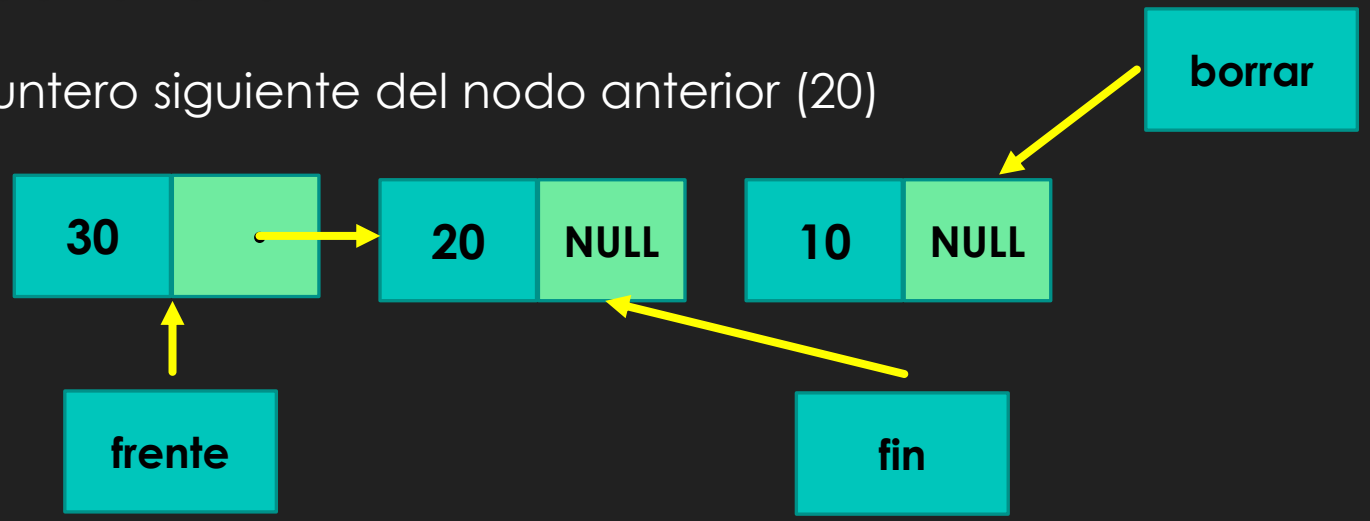


3. Colas

Borrar elementos (3)

- Sea una cola que contiene los elementos 10, 20 y 30
- Queremos borrar el elemento 10

Paso 3: modificar el puntero siguiente del nodo anterior (20)



3. Colas

Borrar elementos (4)

- Sea una cola que contiene los elementos 10, 20 y 30
- Queremos borrar el elemento 10

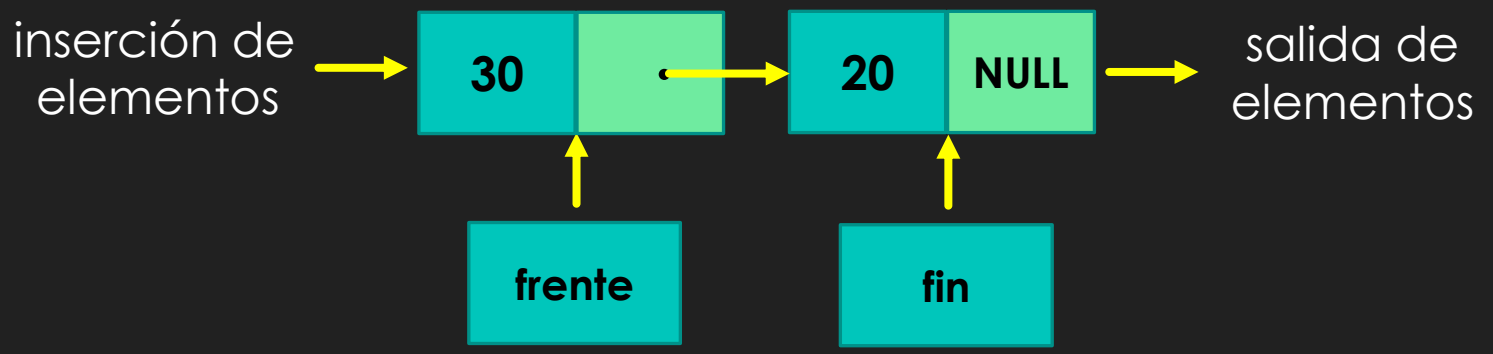
Paso 4: utilizar el valor del nodo a borrar y borrar el nodo

3. Colas

Borrar elementos (5)

- Sea una lista que contiene los elementos 10, 20 y 30
- Queremos borrar el elemento 10

Cola final



Ficheros de ejemplo

Ficheros de ejemplo del tema

- Abre los siguientes ficheros de ejemplo:
  - **12_01_listas.cpp**
  - **12_02_pilas.cpp**
  - **12_03_colas.cpp**

Créditos de las imágenes y figuras

Cliparts e iconos

- **Obtenidos mediante la herramienta web IconFinder** (según sus disposiciones):
  - Diapositiva 1
  - Según la plataforma IconFinder, dicho material puede usarse libremente (free comercial use)
  - A fecha de edición de este material, todos los cliparts son free for comercial use (sin restricciones)

Resto de diagramas y gráficas

- Se han desarrollado en PowerPoint y se han incrustado en esta presentación
- Todos estos materiales se han desarrollado por el autor
  - Si se ha empleado algún icono externo, este se rige según lo expresado anteriormente